**CLAIMS**

What is claimed is:

1. A method of operating a computer having a pipelined processor having a branch target buffer (BTB) comprising creating a recent entry queue in parallel with the branch target buffer (BTB).

2. The method of claim 1 wherein the recent entry queue comprises a set of branch target buffer (BTB) entries.

3. The method of claim 2 comprising organizing the recent entry queue as a FIFO queue.

4. The method of claim 3 wherein the recent entry queue is full associative for reading.

5. The method of claim 1 comparing an entry to be written into the BTB against the valid entries within the recent entry queue.

6. The method of claim 5 comprising blocking an entry matching an entry within the recent entry queue from being written into the BTB.

7. The method of claim 5 wherein when an entry is written into the BTB it is also written into the recent entry queue.

8. The method of claim 1 comprising searching the BTB for a next predicted branch and evaluating the recent entry queue while the BTB is being indexed.

9. The method of claim 8 wherein the recent entry queue maintains a depth up to the associativity of the BTB whereby while the BTB is indexed, the recent entry queue positions are input to comparison logic.

10. The method of claim 8 comprising searching the recent entry queue depth in respect to a matching branch in parallel to searching BTB output.

11. The method of claim 10 comprising creating hit detect logic to supports the associativity of the BTB.

12. The method of claim 8 comprising using a subset of the recent entry queue as a subset of the BTB.

13. The method of claim 12 comprising fast indexing recently encountered branches.

14. The method of claim 12 searching the complete recent entry queue to block duplicate BTB writes.

15. The method of claim 1 comprising searching the recent entry queue to detect looping branches.

16. The method of claim 15 comprising comparing the branch to determine if it was recently written into the queue.

17. The method of claim 16 determining if the branch is backwards branching whereby a looping branch is detected.

18. The method of claim 17 comprising first detecting a looping branch is detected that is not predicted, and thereafter delaying a decode.

19. The method of claim 18 comprising delaying decode until a fixed number of cycles.

20. The method of claim 19 comprising delaying decode until the BTB predicts a branch.

21. The method of claim 1 comprising staging writes to the BTB in the recent entry queue.

22. The method of claim 21 delaying a write and placing the write in the recent event queue.

23. The method of claim 22 detecting a predicted branch while its BTB write is temporarily staged in the recent entry queue.

24. A computer having a pipelined processor comprising a branch target buffer (BTB) comprising with a recent entry queue in parallel with the branch target buffer (BTB).

25. The computer of claim 24 wherein the recent entry queue comprises a set of branch target buffer (BTB) entries.

26. The computer of claim 25 wherein the recent entry queue is a FIFO queue.

27. The computer of claim 26 wherein the recent entry queue is full associative for reading.

28. A program product comprising computer readable code for controlling and configuring a computer having a pipelined processor and a branch target buffer (BTB) to creating a recent entry queue in parallel with the branch target buffer (BTB).

29. The program product of claim 28 wherein the recent entry queue comprises a set of branch target buffer (BTB) entries.

30. The program product of claim 29 further comprising code for organizing the recent entry queue as a FIFO queue.

31. The program product of claim 30 comprising code for making the recent entry queue is full associative for reading.

32. The program product of claim 28 further comprising code for comparing an entry to be written into the BTB against the valid entries within the recent entry queue.

33. The program product of claim 32 further comprising code for blocking an entry matching an entry within the recent entry queue from being written into the BTB.

34. The program product of claim 32 further comprising code for writing an entry into the recent entry queue when the entry is written into the BTB.

35. The program product of claim 28 comprising code for searching the BTB for a next predicted branch and evaluating the recent entry queue while the BTB is being indexed.

36. The program product of claim 35 comprising code for maintaining the recent entry queue at a depth up to the associativity of the BTB whereby while the BTB is indexed, where the recent entry queue positions are input to comparison logic.

37. The program product of claim 35 comprising code for searching the recent entry queue depth in respect to a matching branch in parallel to searching BTB output.

38. The program product of claim 37 comprising code for creating hit detect logic to support the associativity of the BTB.

39. The program product of claim 35 comprising code for using a subset of the recent entry queue as a subset of the BTB.

40. The program product of claim 39 comprising code for fast indexing recently encountered branches.

41. The program product of claim 39 code for searching the complete recent entry queue to block duplicate BTB writes.

42. The program product of claim 28 comprising code for searching the recent entry queue to detect looping branches.

43. The program product of claim 42 comprising code for comparing the branch to determine if it was recently written into the queue.

44. The program product of claim 43 code for determining if the branch is backwards branching whereby a looping branch is detected.

45. The program product of claim 44 comprising code for first detecting a looping branch is detected that is not predicted, and thereafter delaying a decode.

46. The program product of claim 45 comprising code for delaying decode until a fixed number of cycles.

47. The program product of claim 46 comprising code for delaying decode until the BTB predicts a branch.

48. The program product of claim 28 comprising code for staging writes to the BTB in the recent entry queue.

49. The program product of claim 48 code for delaying a write and placing the write in the recent event queue.

50. The program product of claim 49 code for detecting a predicted branch while its BTB write is temporarily staged in the recent entry queue.

* * * * *